REMARKS

Claims 1-35 are currently pending in the application. Claims 1, 9, 18, and 33-35 have been amended in the current response. Applicants have amended the title and abstract of the application in an effort to make them less objectionable to Examiner. Fig. 9 has been replaced to correct a typographical error.

The Examiner rejected claims 1-35 under 35 U.S.C. 103 as being unpatentable over Nilsen (US Patent 6,081,665) in view of Belkin (US Patent 6,542,920). Applicants traverse this rejections because the references in combination do not disclose or suggest the invention as claimed.

Each of independent claims 1 and 33-34, as amended, discloses two classes of code, each associated with a distinct stack memory requirement, through its recital of a "first class of code associated with a first pre-determined stack memory requirement" and a "second class of code associated with a second pre-determined stack memory requirement that is greater than the first stack memory requirement". Each of the two classes of code is *pre-determined* to be associated with a specific stack memory requirement. This enables memory to be allocated appropriately *before* execution of a code segment, thereby allowing stack resources to be conserved. Similarly, claims 9 and 35 each recite a "first class of function calls associated with a first pre-determined stack memory requirement" and "second class of function calls associated with a second pre-determined stack memory requirement that is greater than the first stack memory requirement." Claim 18 discloses, "default stack memory" and "auxiliary stack memory", and "prior to execution of the function calls of the program code, identifying function calls of the program code requiring stack memory greater than the default stack memory."

Nilsen has no such "pre-determined stack memory requirement" associated with classes of codes or function calls. On the contrary, Nilsen's memory requirements are determined "on the fly" *after* execution of the code has begun:

> "In general, each run-time stack is allowed to expand on the fly as necessary. Expansion occurs whenever a stack overflow is detected. Expansion consists of allocating a new stack segment, copying that portion of the original stack segment that is necessary to

> establish an execution context on the new stack segment (the incoming parameters, for example), adjusting links to represent the addition of the new stack segment and setting the corresponding stack pointer(s) to their new values." (emphasis added) (36:56-64)

In Nilsen, a "new stack segment" is allocated only when the triggering event of "a stack overflow" takes place. (Id.) This is a completely different way of assigning memory than is claimed. In claims 1, 9 and 33-35, as amended, classes of code or function calls are "associated with a [] pre-determined stack memory requirement" prior to execution of the function calls of the program code. Similarly, Claim 18 discloses, "prior to execution of the function calls of the program code, identifying function calls of the program code requiring stack memory greater than the default stack memory." The memory requirement associated with a code segment or function call is "pre-determined" or identified "prior to execution" as claimed. If execution of a code segment is pre-determined to require extra memory, the code executes in an "auxiliary stack memory" (claims 1, 9, 18, 34, and 35) or "second stack memory" (Claim 33). This predetermined association of the code classes with memory eliminates the need to detect stack overflow, for example. On the contrary, in Nilsen, only if there is a "stack overflow" is a "new stack segment" added "on the fly." Nowhere does Nilsen disclose or suggest associating classes of code with "pre-determined memory requirement[s]". In Nilsen, if a task requires extra memory, a new stack segment is assigned during, not before, execution.

Belkin does not overcome this deficiency. Belkin describes "thread pools," and a "request" that is "processed to determine with which thread pool the request is to be associated". (Abstract) Belkin does not disclose classes of code or classes of function calls are "associated with a [] pre-determined stack memory requirement" prior to execution of the function calls or program code. Nor does it disclose "prior to execution of the function calls of the program code, identifying function calls of the program code requiring stack memory greater than the default stack memory" as claimed. At best it discloses a thread pool with characteristics including "a stack size." (5:65- 6:3) But classes of code are not " associated with a [] pre-determined stack memory requirement" as claimed. Even later, when a request is associated with a thread pool,

Belkin discloses making a determination based on "whether the request is requesting a functionality that has a specific thread pool associated therewith" or "making a determination as to the type of service that is being requested by the request," (9:17-36) not making a determination based on a stack memory requirement.

At best, the combination of the teachings of the references merely results in associating a request with a thread pool depending on the functionality or type of service requested, and then making additional stack resources available if a stack overflow is detected. This does not meet the claimed invention. For at least these reasons, neither the cited references alone or in combination disclose or suggest the subject matter of claims 1, 9, 18, and 33-35, and by extension dependent claims 2-8, 10-17, and 19-32.

In addition, each of claims 2-4 discloses a "wrapper configured to call the auxiliary stack memory to execute the function call", each of claim 13-15 discloses a "wrapping the program code in a wrapper to transfer the execution to the auxiliary stack memory", and each of claims 18-32 discloses "wrapping each function call ...with a wrapper configured to call an auxiliary stack memory to execute the function call." Neither of the cited references disclose a "wrapper" (indeed, a word search of both specifications resulted in no hits of the word "wrapper") much less a wrapper "configured to call [] [the] auxiliary stack memory" or "to transfer the execution to the auxiliary stack memory" as claimed. In contrast, at best, in Nilsen, "each run-time stack is allowed to expand on the fly as necessary" (36:56-57). In Belkin, the pool with which to associate a request is determined based on "indication information" extracted from the request. (Abstract) Quite clearly, neither reference discloses the subject matter of claims 2-4, 13-15, and 18-32.

Accordingly, Applicants submit that the claims are patentably distinct over the cited art. Consideration of this application and the early allowance of all claims herein is requested.

Should the Examiner wish to discuss the above amendments and remarks, or if the Examiner believes that for any reason direct contact with Applicants' representative would help to advance the prosecution of this case to finality, the Examiner is invited to telephone the undersigned at the number given below.

Respectfully submitted,
HONG ZHANG *et al.*

Dated: March 4, 2004

By: _____

Colleen V. Chien, Reg. No. 55,062
Attorney for Applicants
Fenwick & West LLP
Silicon Valley Center
801 California Street
Mountain View, CA 94041
Tel.: (415) 875-2319
Fax: (415) 281-1350